Codes for Multi-Level Flash Memories: Correcting Asymmetric Limited-Magnitude Errors

Yuval Cassuto **Moshe Schwartz** **Vasken Bohossian** **Jehoshua Bruck**
California Institute of Technology
1200 E California Blvd., Mail Code 136-93
Pasadena, CA 91125, U.S.A.
{*ycassuto,moosh,vincent,bruck*}*@paradise.caltech.edu*

***Abstract*— Several physical effects that limit the reliability and performance of Multilevel Flash memories induce errors that have low magnitude and are dominantly asymmetric. This paper studies block codes for asymmetric limited-magnitude errors over q-ary channels. We propose code constructions for such channels when the number of errors is bounded by t. The construction uses known codes for symmetric errors over small alphabets to protect large-alphabet symbols from asymmetric limited-magnitude errors. The encoding and decoding of these codes are performed over the small alphabet whose size depends only on the maximum error magnitude and is independent of the alphabet size of the outer code. An extension of the construction is proposed to include systematic codes as a benefit to practical implementation.**

I. Introduction

Flash Memory is a Non-Volatile Memory (NVM) technology that is both electrically programmable and electrically erasable. This property, together with high storage densities and high speed programming, has made Flash Memory the dominant non-volatile memory technology and a prominent enabler for many portable applications and technologies. The Flash Memory market is estimated in the tens of billions of dollars, and it has a huge growth potential, some of it at the expense of volatile memories and magnetic storage media.

Hand in hand with opportunity, come the challenges of designing reliable Flash memories with higher storage volumes and lower costs per byte. At the current state of matters, the most efficient way to scale the storage density of Flash memories, is to use the *Multi-Level Flash Cell* concept to increase the number of stored bits in a cell [5] (and references therein). Contrary to ubiquitous single bit Flash memories, where each cell is in one of two (Erased/Programmed) threshold states, Multilevel Flash memories use a state space of 2^b threshold levels, to store b bits in a single cell. Since physical/engineering factors limit the overall window of threshold levels, an obvious consequence of the Multilevel Flash concept is both a requirement for fast accurate charge placement mechanisms and compromised reliability margins (that lead to errors in stored date) [4]. The physical processes that introduce errors typically move cells to adjacent threshold levels, commonly in one dominant direction. Next we elaborate on these phenomena to motivate the focus of this paper on codes for *limited-magnitude, asymmetric* errors.

Being the paramount challenge of Multilevel Flash memory implementation, fast accurate program schemes are a topic of significant research and design efforts [8],[2],[6]. All these and other works, share the attempt to iteratively program a cell to an *exact* prescribed level, in a minimal number of program cycles. It is well known that Flash memory technology does not support charge removal from *individual* cells. As a result, the program cycle sequence is designed to "cautiously" approach the target value from below, to avoid undesired global erases in cases of overshoots. Consequently, these attempts still require many program cycles (order of 10 or more) and they work only up to a moderate number of bits per cell. A key observation that motivated this work, is that if program overshoots can be tolerated, then more "aggressive" program cycles can be employed to yield significantly faster memory write operations. The way to contain the (controlled) inaccuracies resulting from these overshoots, is by devising error-correcting codes, with *specific* properties to combat errors that originate from the program process. The most appropriate model to capture these errors is by assuming channel errors that have limited magnitude and are asymmetric. The limited-magnitude assumption stems from the employment of programming schemes that can guarantee some level of accuracy. The asymmetry of the errors is due to the property that all errors are overshoots of the desired target value.

Besides the need for accurate programming, the move to Multilevel Flash cells also aggravates reliability concerns in the design and operation of Flash memories. The same reliability aspects (that were successfully handled in Single Level Flash memories), may become more significant and translate to errors in stored data. Many of these potential errors also motivate the asymmetric limited-magnitude error channel. Low *data retention* [3], caused by a slow loss of charge from memory cells, is one such example. Another is errors that originate from low *memory endurance*, by which a drift of threshold levels in aging devices [3] may cause program and read errors. Program and read *disturbs* [3], caused by programming/reading proximate memory cells, also induce low magnitude errors with a dominant direction of change.

In this paper we study block codes for asymmetric limited-

This work was supported in part by the Caltech Lee Center for Advanced Networking.

1-4244-1429-6/07/$25.00 ©2007 IEEE

magnitude errors. The codes are parametrized by ℓ, the maximum magnitude of an error, and t, the maximum number of asymmetric limited-magnitude errors in a codeword of length n. Asymmetric limited-magnitude error-correcting codes were recently proposed in [1] for the case $t = n$. These codes turn out to be a special case of the general construction method detailed below.

The following example illustrates the coding problem and introduces the main idea of the code construction. Suppose we have a group of 5 cells, each in one of 8 possible threshold levels, marked by the integers $\{0, 1, \ldots, 7\}$. The design goal is now chosen to be, protecting this group of cells against $t = 2$ errors of magnitude $\ell = 1$ in the upward direction. As illustrated by the sample words below, if the stored levels are restricted to have either all symbols with even parity or all symbols with odd parity, the required protection is achieved. For each of the two sample codewords in row (a) of the figure below, the channel introduces two upward errors of magnitude 1 (b). By majority, the locations of the errors are detected (c), in bold, and the original symbols are recovered by decrementing the erroneous symbols (d). The

	Sample 1					Sample 2				
	codeword					codeword				
(a)	3	5	3	1	1	4	6	2	2	0
	corrupted					corrupted				
(b)	4	5	3	2	1	4	6	3	2	1
	detected					detected				
(c)	**4**	5	3	**2**	1	4	6	**3**	2	**1**
	corrected					corrected				
(d)	**3**	5	3	**1**	1	4	6	**2**	2	**0**

example above is one instantiation of a general construction method that provides codes for all possible code parameters. The main strength of this method is that for any target alphabet size (determined by the number of threshold levels), asymmetric limited-magnitude error correctability is inherited from *symmetric* error correctability of codes over alphabets of size $\ell + 1$ (in the case of the example above, it is the binary repetition code.). Thus a rich selection of known symmetric-error-correcting codes becomes handy to offer codes that are optimized for the asymmetric limited-magnitude channel. As a favorable by-product of the construction method, encoding and decoding of the resulting codes are performed on symbol sets whose sizes depend only on ℓ, irrespective of the code alphabet (which may be much larger than ℓ). This is a major advantage in both redundancy and complexity, compared to other proposed codes for Multilevel Flash memories (e.g [7]), whose encoding and decoding are performed over the large code alphabet. Following the definition of the coding problem and the presentation of the non-systematic code construction in section II, we demonstrate its power by showing that it provides code families that are perfect in the asymmetric limited-magnitude sense. We note that the general idea of the basic code construction, restricted to binary codes, has appeared in Construction A of [9], for a different, though related, application (sphere packings in Euclidean spaces). In later sections, we refine and modify the basic code construction to attain useful properties for practical implementations. Section IV discusses different possible mappings from information symbols to code symbols. Section V provides general constructions for *systematic* codes, that are advantageous in high-speed memory architectures. Finally, extensions and future research opportunities are discussed.

II. t Asymmetric ℓ-Limited-Magnitude Error-Correcting Codes

An alphabet Q of size q is defined as the set of integers modulo q: $\{0, 1, 2, \ldots, q-1\}$. For a codeword $\boldsymbol{x} \in Q^n$ and a channel output $\boldsymbol{y} \in Q^n$, the definition of asymmetric limited-magnitude errors now follows.

Definition 1. *Given a codeword $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in Q^n$ and a channel output $\boldsymbol{y} = (y_1, y_2, \ldots, y_n) \in Q^n$, we say that t AℓM (Asymmetric ℓ-limited-Magnitude) errors occurred if $|\{i : y_i \neq x_i\}| = t$, and for all i, $y_i \geqslant x_i$ and $y_i - x_i \leqslant \ell$. A generalization of the above definition is when we allow asymmetric errors to wrap around (from $q-1$ back to 0). We say that t AℓM errors* with wrap-around *occurred if $|\{i : y_i \neq x_i\}| = t$, and for all i, $(y_i - x_i) \bmod q \leqslant \ell$.*

For notational convenience, given $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, the vector $(x_1 \bmod q', x_2 \bmod q', \ldots, x_n \bmod q')$ will be denoted by $\boldsymbol{x} \bmod q'$.

The discussion of codes for this channel model is commenced with the definition of a distance that captures the correctability of t AℓM errors.

Definition 2. *For $\boldsymbol{x} = (x_1, \ldots, x_n) \in Q^n$ and $\boldsymbol{y} = (y_1, \ldots, y_n) \in Q^n$, define $N(\boldsymbol{x}, \boldsymbol{y}) = |\{i : x_i > y_i\}|$ and $N(\boldsymbol{y}, \boldsymbol{x}) = |\{i : x_i < y_i\}|$. The distance d_ℓ between the words $\boldsymbol{x}, \boldsymbol{y}$ is defined*

$$d_\ell(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} n+1 & \text{if } \max_i(|x_i - y_i|) > \ell \\ \max(N(\boldsymbol{x}, \boldsymbol{y}), N(\boldsymbol{y}, \boldsymbol{x})) & \text{otherwise} \end{cases}$$

The d_ℓ distance defined above allows to determine the number of AℓM errors, correctable by a code $\mathcal{C}$. (proof omitted)

Proposition 3 *A code $\mathcal{C} \subset Q^n$ can correct t AℓM errors if and only if $d_\ell(\boldsymbol{x}, \boldsymbol{y}) \geqslant t + 1$ for all distinct $\boldsymbol{x}, \boldsymbol{y}$ in $\mathcal{C}$.*

To prove the correctability properties of the codes we soon propose, we do not resort to Proposition 3 above. Rather, a stronger way of proving correctability is used: providing decoding algorithms that use properties of known codes to guarantee successful decoding. We now provide the main construction of the paper. To obtain a code over alphabet Q that corrects t or less AℓM errors, one can use codes over smaller alphabets as follows. Let Σ be a code over the alphabet Q' of size q'. The code $\mathcal{C}$ over the alphabet Q of size q $(q > q' > \ell)$ is defined as

$$\mathcal{C} = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in Q^n : \boldsymbol{x} \bmod q' \in \Sigma\}. \quad (1)$$

Error-correction properties of $\mathcal{C}$ are derived from those of Σ in the following manner.

Theorem 4. *$\mathcal{C}$ corrects t AℓM errors if Σ corrects t asymmetric ℓ-limited-magnitude errors with wrap-around. If $q \geqslant q' + \ell$,[1] the converse is true as well.*

Proof: Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{C}$ be a codeword and $\boldsymbol{y} = (y_1, y_2, \ldots, y_n) \in Q^n$ be the channel output when t AℓM errors have occurred. Denote the corresponding Σ codeword by $\boldsymbol{\chi} = \boldsymbol{x} \bmod q'$, and also define $\boldsymbol{\psi} = \boldsymbol{y} \bmod q'$ and $\boldsymbol{\epsilon} = (\boldsymbol{\psi} - \boldsymbol{\chi}) \pmod{q'}$. First we observe that since $q' > \ell$, if $0 \leqslant y_i - x_i \leqslant \ell$ then $y_i - x_i = (y_i - x_i) \bmod q'$. Using the simple modular identity

$$\begin{aligned}(y_i - x_i) \bmod q' &= (y_i \bmod q' - x_i \bmod q') \bmod q' \\ &= (\psi_i - \chi_i) \bmod q' = \epsilon_i,\end{aligned}$$

we get that $y_i - x_i = \epsilon_i$, and in particular, if $0 \leqslant y_i - x_i \leqslant \ell$, then $0 \leqslant \epsilon_i \leqslant \ell$. In other words, if the codeword $\boldsymbol{x}$ over Q suffered an AℓM error at location i, then the codeword $\boldsymbol{\chi}$ over Q' suffered an AℓM error with wrap-around at the same location i, and with the same magnitude. Given at most t AℓM errors with wrap-around, a decoder for Σ can recover $\boldsymbol{\epsilon}$ from $\boldsymbol{\psi}$. Thus, the equality $y_i - x_i = \epsilon_i$ allows the same decoder to recover $\boldsymbol{x}$ from $\boldsymbol{y}$. The converse is settled by observing that, for $q \geqslant q' + \ell$, a non-correctable error $\boldsymbol{\epsilon}$ for Σ can be used to generate a non-correctable $\boldsymbol{y}$ vector for $\mathcal{C}$. ■

Remark: If $q' \mid q$ then $\mathcal{C}$ corrects t AℓM errors *with wrap-around* for Σ with the same properties as above.

The size of the code $\mathcal{C}$ is bounded from below and from above by the following theorem. (proof omitted)

Theorem 5. *The number of codewords in the code $\mathcal{C}$ is bounded by the following inequalities.*

$$\left\lfloor \frac{q}{q'} \right\rfloor^n \cdot |\Sigma| \leqslant |\mathcal{C}| \leqslant \left\lceil \frac{q}{q'} \right\rceil^n \cdot |\Sigma|$$

In the special case when $q' = 2$, the size of $\mathcal{C}$ can be obtained exactly from the weight enumerator of Σ. (proof omitted)

Theorem 6. *Let $q' = 2$ and Σ be a code over $Q' = \{0, 1\}$. Then the size of the code $\mathcal{C}$, as defined in (1), is given by*

$$|\mathcal{C}| = \sum_{w=0}^{n} A_w \left\lceil \frac{q}{2} \right\rceil^{n-w} \left\lfloor \frac{q}{2} \right\rfloor^{w}$$

where A_w is the number of codewords of Σ with Hamming weight w.

This theorem can be extended to $q' > 2$, but in such cases knowing the weight distribution of Σ does not suffice, and more detailed enumeration of the code is needed for an exact count.

While (1) provides a fairly general way of composing t AℓM error-correcting codes from similar codes over smaller alphabets, it is the following special case of this composition method, that proves most useful for obtaining strong and efficient codes.

Let Σ be a code over the alphabet Q', now of size $\ell + 1$. The code $\mathcal{C}$ over the alphabet Q of size q ($q > \ell + 1$) is defined as

$$\mathcal{C} = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in Q^n : \boldsymbol{x} \bmod (\ell + 1) \in \Sigma\}. \quad (2)$$

[1] a reasonable assumption since the best codes are obtained when $q >> q'$

In this special case $\mathcal{C}$ has the following property.

Theorem 7. *$\mathcal{C}$ corrects t AℓM errors if and only if Σ corrects t symmetric errors.*

Proof: When $q' = \ell + 1$, an AℓM error with wrap-around is equivalent to a symmetric error. This is therefore, a special case of Theorem 4. ■

The ℓ-AEC codes suggested in [1], that correct all AℓM errors, can also be regarded as a special case of this construction method. To show that, let $\mathbf{0}$ be the trivial length n code, over the alphabet Q' of size $\ell + 1$, that contains only the all-zero codeword. Define

$$\begin{aligned}\mathcal{C} &= \{\boldsymbol{x} \in Q^n : \boldsymbol{x} \bmod (\ell + 1) \in \mathbf{0}\} \\ &= \{\boldsymbol{x} \in Q^n : x_i \equiv 0 \bmod (\ell + 1) \text{ for } i = 1, 2, \ldots, n\} \text{ [1].}\end{aligned}$$

Since $\mathbf{0}$ can correct $t = n$ symmetric errors, $\mathcal{C}$ can correct $t = n$ AℓM errors.

III. Perfect Asymmetric Limited-Magnitude Codes

To showcase the power of the code construction from section II, we demonstrate how it can yield perfect codes in the AℓM error model. For that we first give (without proof) a generalization of the q-ary symmetric "sphere packing" bound to the case of asymmetric limited-magnitude errors. We then show that asymmetric limited-magnitude error-correcting codes that meet this bound can be obtained by known perfect codes in the Hamming metric.

Theorem 8. *If $\mathcal{C}$ is a t AℓM error-correcting code with wrap-around, of length n over an alphabet of size q, then*

$$|\mathcal{C}| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i \leqslant q^n$$

Perfect t AℓM error-correcting codes are obtained through the following proposition. (proof omitted)

Proposition 9 *If there exists a perfect (in the Hamming metric) code over an alphabet of size $\ell + 1$, then there exists a perfect AℓM code with the same length, over an alphabet of any size q, such that $\ell + 1 \mid q$, that corrects the same number of errors.*

IV. Encoding and Decoding of AℓM Codes

The method of code construction proposed in (1), specified the code $\mathcal{C}$ as a subset of Q^n. Moreover, the proof of Theorem 4 implicitly contained a decoding algorithm for $\mathcal{C}$, given a decoder for the smaller-alphabet code. Nevertheless, till this point, no encoding function from information symbols to codewords was provided. Discussing this encoding function is crucial when a practical coding scheme is required. When $q' \mid q$, a straightforward encoding function from information symbols over Q' to codewords of $\mathcal{C}$ over Q exists. We show this function using the following example.

Example 1. *Let Σ_H be the binary[2] Hamming code of length $n = 2^m - 1$, for some integer m. First we define the code $\mathcal{C}_H$ in the way of section II.*

$$\mathcal{C}_H = \{\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in Q^n : \boldsymbol{x} \bmod 2 \in \Sigma_H\}.$$

[2] Non-binary Hamming codes can be used as well when $\ell > 1$.

By the properties of Σ_H, the code $\mathcal{C}_H$ corrects one A1M error. When the code alphabet size is $q = 2^b$, for some integer b, the perfect code $\mathcal{C}_H$ admits a simple function from $nb - m$ information bits to codewords of $\mathcal{C}_H$ over Q. A possible encoding scheme is illustrated in Figure 1 below. In Figure 1 (a), $nb - m$ information bits are input to the encoder. The encoder then uses a binary Hamming encoder to encode $n - m$ of the information bits into a length n Hamming codeword (Figure 1 (b)). Finally, in Figure 1 (c), each q-ary symbol of the codeword $\boldsymbol{x} \in \mathcal{C}_H$ is constructed from b bits using the usual binary-to-integer conversion, the top row being the least-significant bits of $x_i \in Q$. Decoding is carried out by using a Hamming decoder

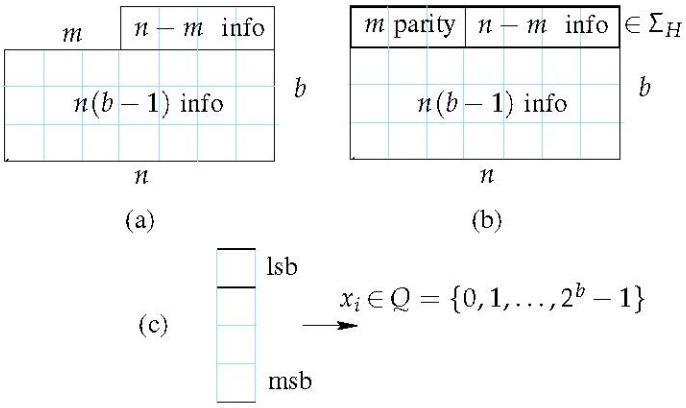

Figure 1. Encoding Procedure for $\mathcal{C}_H$

to find the limited-magnitude error location and magnitude (for binary Hamming codes the magnitude is always 1). This value is then subtracted from the word $\boldsymbol{y}$ to obtain a decoded codeword. To recover the information bits after decoding, the Q symbols are converted back to bits in the usual way, and the m parity bits are discarded.

In Example 1, a simple encoding function for $\mathcal{C}$ was given in conjunction with a mapping from tuples of b bits to symbols of Q. Alternative pairs of encoding functions and symbol mappings can be found for the same code, that give better bit error probability when the code fails to correct all errors. One such example given below, is a hybrid between the usual positional mapping and the Gray mapping.

1) The hybrid mapping: Given a Gray code of length $b-1$, if the length $b-1$ bit string $a_{b-1} \cdots a_1$ has sequential number α in this Gray code, then the b-tuple is mapped to the number $2\alpha + a_0$. This mapping has the advantage that it can still be used in conjunction with the simple least-significant bit encoding, and in addition, it reduces the bit error probability when errors with magnitude greater than ℓ occur.

V. Systematic Codes

All their advantages notwithstanding, the codes discussed thus far in the paper suffer the shortcoming of not admitting a systematic representation over Q. As seen in Figure 1(b), $(b-1)m$ of the information bits are encoded in the m symbols that also carry parity bits. Symbols that carry both information and parity bits are undesirable if the code is used in high speed memory applications, in which fast reading and writing is a requirement. A trivial and wasteful way to obtain systematic codes from the construction above is by not using these problematic $(b-1)m$ information bits. This section aims at finding more clever constructions for systematic codes.

A. Systematic Codes for $\ell = 1$ Limited-Magnitude Errors

When the error magnitude ℓ is bounded by 1, the code Σ in the code construction (2) is a binary code. As we show next for this case, a modification of any code $\mathcal{C}$ can be carried out, that yields a (slightly shorter) systematic code with the same correction capability. We start off with an example.

Example 2. *In this example we propose a systematic variant to the code $\mathcal{C}_H$, given in Example 1. The encoding function given below generates a code that has the same correction capabilities as $\mathcal{C}_H$, namely any single $\ell = 1$ asymmetric error is correctable, though the resulting code is different. Specifically, the dimensions of the systematic code are different. For this example we assume that the alphabet size of the code is 2^m (m – the number of parity bits in the binary code), compared to 2^b for arbitrary b in $\mathcal{C}_H$. This assumption can be lifted with a small increase in redundancy that depends on the actual code parameters. For an $[n, k = n - m]$ binary Hamming code Σ_H, the length of the systematic code is $n - m + 1$, compared to n in the non-systematic case. The systematic code is encoded as follows. In Figure 2 (a), km information bits are input to the encoder. The encoder then uses a binary Hamming encoder to encode the k information bits of the top row into a length $n = k + m$ Hamming codeword (Figure 2 (b)). The parity bits of the Hamming codeword are now placed as a separate column. The mapping of bits to Q symbols, shown in Figure 2 (c), is the usual positional mapping for the k information symbols and the Gray mapping for the parity symbol.*

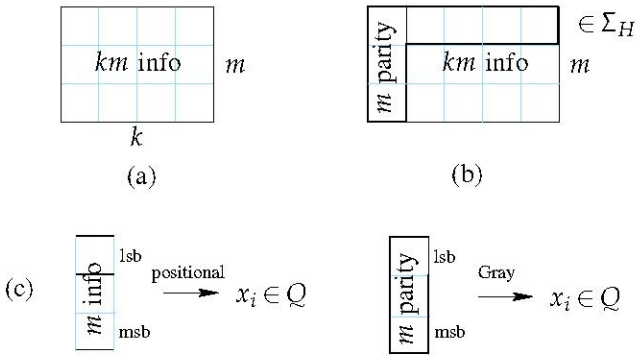

Figure 2. Encoding Procedure for a Systematic Code

To decode, a word of Q^{k+1} is converted back to bits using the same mappings, and a binary Hamming decoder is invoked for the n coded bits. By construction, a single $\ell = 1$ asymmetric error over Q, translates to a single bit error in the Hamming codeword: in the k information symbols, an $\ell = 1$ error flips a single (coded) least-significant bit (and perhaps other bits in the same column), and in the parity symbol, an $\ell = 1$ error flips exactly one parity bit in that column, thanks to the Gray code used in the mapping.

The code proposed in Example 2, together with its encoding/decoding, can be generalized to any t A1M error-correcting code as stated by the following proposition (proof omitted).

Proposition 10 *Let Σ be a binary systematic code of length n and $m \leqslant b \cdot r$ parity bits, for any two integers r and $b > 1$. If Σ corrects t symmetric errors, then it can be used to construct a* systematic *t A1M error-correcting code over an alphabet of size $q = 2^b$. This code has length $n - m + r$, of which r symbols are parity symbols.*

B. Systematic Codes for $\ell > 1$ Limited-Magnitude Errors

If we try to extend the construction of the previous subsection to codes for asymmetric $\ell > 1$ limited-magnitude errors, we immediately face a stumbling block. Although generalized Gray codes exist for non-binary alphabets, they (like all other mappings) do not guarantee that a single AℓM error translates to a single symmetric error in the $(\ell+1)$-ary codeword. We next consider ways to overcome this difficulty to reattain the generality of the systematic construction. The first proposed solution is simple but wasteful, the second is significantly more efficient for large ℓ.

1) Making Parity Symbols Error-Free: If the parity Q symbols are taken from a subset of Q with relative size $1/(\ell+1)$, AℓM errors in the parity symbols can be easily corrected before invoking the decoder for Σ. This solves the problem at the cost of one $(\ell+1)$-ary symbol per q-ary parity symbol.

2) Ensuring a Single Symmetric Error with a Small Added Redundancy: When we examine more closely the properties of the $(\ell+1)$-ary reflected Gray code, we see that for any ℓ and b, an AℓM error induces at most *two* symmetric errors in the $(\ell+1)$-ary code. Moreover, if an AℓM error induces two symmetric errors, then one of the two has to be in the right most location of the Gray codeword. While these properties themselves are not satisfactory for the construction, with a small amount of added redundancy (that becomes negligible with increasing ℓ), one can guarantee that at most one $(\ell+1)$-ary symbol is changed by the AℓM error. We first define the well known N-ary reflected Gray code and prove its aforementioned properties.

Definition 11. *For an even N, let the N-ary reflected Gray code of length b be defined recursively, as follows.*

$$G(1,N) = \begin{pmatrix} 0 \\ 1 \\ 2 \\ \vdots \\ N-1 \end{pmatrix}, \; G(b,N) = \begin{pmatrix} \mathbf{0} & G(b-1,N) \\ \mathbf{1} & G(b-1,N)^R \\ \mathbf{2} & G(b-1,N) \\ \vdots & \vdots \\ \mathbf{N-1} & G(b-1,N)^R \end{pmatrix}$$

where the symbols in bold represent column vectors with N^{b-1} identical elements. The sub-matrix $G(b-1,N)^R$ stands for $G(b-1,N)$ in reversed order.

Theorem 12. *Let two words of $G(b,N)$ be denoted $x_{j_1} = a_{b-1}a_{b-2}\cdots a_2a_1a_0$ and $x_{j_2} = c_{b-1}c_{b-2}\cdots c_2c_1c_0$, respectively. If $|j_2 - j_1| \leqslant N$, then $D(a_{b-1}a_{b-2}\cdots a_2a_1, c_{b-1}c_{b-2}\cdots c_2c_1) \leqslant 1$. $D(\boldsymbol{x},\boldsymbol{y})$ is the Hamming distance between two words. Moreover, a_i and c_i on the differing location i, differ by exactly 1 modulo N.*

Proof: By construction, the i^{th} symbol has ± 1 transitions in list numbers $j = sN^i$, where $s \equiv 1, \ldots, N-1 \pmod N$. In particular, for $i > 0$, $N \mid j$ and therefore transitions are at least N apart in the upper $n-1$ indices. ■

The theorem proves that given an AℓM error, at most one of the upper $b-1$ $(\ell+1)$-ary symbols suffers an error with magnitude ± 1 (i.e $c_i \equiv a_i \pm 1 \pmod{\ell+1}$). Consequently, if we use only Gray codewords whose $b-1$ upper symbols have even parity, then an asymmetric $(\ell+1)$-limited-magnitude error induces only a single symmetric error (in the zeroth symbol) and thus the systematic construction for $\ell = 1$ works for a general odd[3] ℓ. This restriction on the contents of the parity symbols of $\mathcal{C}$, amounts to roughly one bit of additional redundancy per q-ary parity symbol. For increasing ℓ, this is a negligible loss in storage efficiency, compared to a full $(\ell+1)$-ary redundant symbol that was required in the solution in V-B.1 above. Ways to map $(\ell+1)$-ary parity symbols into the restricted alphabet of the q-ary parity symbols are omitted in this presentation.

VI. Conclusions and Future Research

Many of the strengths of the code construction method were not explored in the current paper. For example, similar ideas can lead to codes that correct symmetric limited-magnitude errors, or more generally, asymmetric double-sided limited-magnitude errors that may arise in particular designs. Also, when the reading resolution is greater than the code alphabet size, improved decoding techniques can be readily applied using "limited-magnitude erasures" or other soft interpretations of the read symbols. Better systematic codes may be obtained by observing the relation between the limited magnitude errors and the errors they impose on the low-alphabet code, and then replacing the symmetric error correction properties we required (which are too strong) with various Unequal Error Protection properties.

References

[1] R. Ahlswede, H. Aydinian, L. Khachatrian, and L. Tolhuizen, "On q-ary codes correcting all unidirectional errors of a limited magnitude," Arxiv.org, http://arxiv.org/abs/cs/0607132, Tech. Rep. cs.IT/0607132, 2006.

[2] A. Bandyopadhyay, G. Serrano, and P. Hasler, "Programming analog computational memory elements to 0.2% accuracy over 3.5 decades using a predictive method," in *proc. of the IEEE International Symposium on Circuits and Systems*, 2005, pp. 2148–2151.

[3] P. Cappelletti and A. Modelli, "Flash memory reliability," *Flash Memories, P. Cappelletti, C. Golla, P. Olivo, E. Zanoni Eds. Kluwer*, pp. 399–441, 1999.

[4] B. Eitan, R. Kazerounian, and A. Roy, "Multilevel Flash cells and their trade-offs," *IEDM Technical Digest*, pp. 169–172, 1996.

[5] B. Eitan and A. Roy, "Binary and multilevel Flash cells," *Flash Memories, P. Cappelletti, C. Golla, P. Olivo, E. Zanoni Eds. Kluwer*, pp. 91–152, 1999.

[6] H. Nobukata et. al, "A 144-Mb, eight-level NAND Flash memory with optimized pulsewidth programming," *IEEE J. Solid-State Circuits*, vol. 35, no. 5, pp. 682–690, 2000.

[7] S. Gregori, A. Cabrini, O. Khouri, and G. Torelli, "On-chip error correcting techniques for new-generation Flash memories," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 602–616, 2003.

[8] M. Grossi, M. Lanzoni, and B. Ricco, "Program schemes for multilevel Flash memories," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 594–601, 2003.

[9] J. Leech and N. Sloane, "Sphere packing and error-correcting codes," *Canadian J. Math.*, vol. 23, no. 4, pp. 718–745, 1971.

[3] The result extends to odd $\ell+1$, but the focus here is on more practical even sized alphabets